Data and text mining

Variance stabilization and normalization for one-color microarray data using a data-driven multiscale approach

E. S. Motakis¹, G. P. Nason¹,*, P. Fryzlewicz¹ and G. A. Rutter²
¹Department of Mathematics and ²Department of Biochemistry, University of Bristol, Bristol, UK

Received on November 4, 2005; revised on July 4, 2006; accepted on July 21, 2006
Advance Access publication July 28, 2006
Associate Editor: Thomas Lengauer

ABSTRACT
Motivation: Many standard statistical techniques are effective on data that are normally distributed with constant variance. Microarray data typically violate these assumptions since they come from non-Gaussian distributions with a non-trivial mean–variance relationship. Several methods have been proposed that transform microarray data to stabilize variance and draw its distribution towards the Gaussian. Some methods, such as log or generalized log, rely on an underlying model for the data. Others, such as the spread-versus-level plot, do not. We propose an alternative data-driven multiscale approach, called the Data-Driven Haar–Fisz for microarrays (DDHFm) with replicates. DDHFm has the advantage of being ‘distribution-free’ in the sense that no parametric model for the underlying microarray data is required to be specified or estimated; hence, DDHFm can be applied very generally, not just to microarray data.

Results: DDHFm achieves very good variance stabilization of microarray data with replicates and produces transformed intensities that are approximately normally distributed. Simulation studies show that it performs better than other existing methods. Application of DDHFm to real one-color cDNA data validates these results.

Availability: The R package of the Data-Driven Haar–Fisz transform (DDHFm) for microarrays is available in Bioconductor and CRAN.

Contact: g.p.nason@bristol.ac.uk

1 INTRODUCTION

Microarrays, in principle and in practice, are extensions of hybridization-based methods (Southern blots, northern blots, SAGE, etc.), which have been used for decades to identify and locate mRNA and DNA sequences that are complementary to a segment of DNA (Alwin et al., 1977; Velculescu et al., 1995). Microarray technology, in the form of either cDNA or high-density oligonucleotide arrays enables molecular biologists to measure simultaneously the expression level of thousands of genes. In a typical microarray experiment the aim is to compare different cell types, e.g. normal versus diseased cells, in order to identify genes that are differentially expressed in the two cell types.

Typically, microarray data analyses consist of several steps ranging from experimental design to the identification of important genes (for a review on the whole process see Sebastiani and Ramoni, 2003). Gene replication is a crucial design feature as it increases the precision of estimation and permits estimation of measurement variance which enables the significance of the final results to be judged.

Rocke and Durbin (2001) identified that the variance of the raw spot intensities increased with their mean and they modeled those intensities in terms of the two-component model:

\[ Y_i = \alpha + \mu_i \times e^{0.5} + \epsilon_i, \quad i = 1, \ldots, n. \]  

Here, \((Y_i)_{i=1}^n\) are the raw single-color intensities for the \(n\) genes, each assumed to be replicated \(p\) times. Sometimes we will write \(Y_{i,r}\), when we are referring to the \(r\)-th replicate on the \(i\)-th gene \((r = 1, \ldots, p)\). The \(\alpha\) term represents the (common) mean background noise of the \(n\) genes on the array, \(\mu_i\) is the true expression level for gene \(i\), and \(\eta_i\) and \(\epsilon_i\) are the normally distributed error terms with zero mean and variances \(\sigma_\eta^2\) and \(\sigma_\epsilon^2\), respectively. In this way, \(Y = (Y_{i,r})_{i=1}^n\) can be considered as coming from an inhomogeneous process that produces the \(n\) gene intensities with finite but different \(\mu_i,\) finite but different variances.

At low expression levels (i.e. \(\mu_i\) close to 0) the measured expression \(Y_i\) in (1) can be written as \(Y_i \approx \alpha + \epsilon_i\) so that \(Y_i\) is approximately distributed as \(N(\alpha, \sigma_\epsilon^2)\). On the other hand, for large \(\mu_i\),s, the middle term in (1) dominates and \(Y_i\) can be modeled as follows:

\[ Y_i \approx \mu_i e^{0.5}, \]  

with approximate variance

\[ \text{Var}(Y_i) \approx \mu_i^2 S_\eta^2, \]  

where \(S_\eta^2 = e^{0.5}(e^{0.5} - 1)\). For moderate values of \(\mu_i\), \(Y_i\) is modeled as in (1) with variance:

\[ \text{Var}(Y_i) = \mu_i^2 S_\eta^2 + \sigma_\epsilon^2. \]  

From (3) and (4), we observe that the SD of the \(Y_i\) increases linearly with their mean. Such mean–variance dependence, implying the presence of heteroscedastic intensities, is a major problem in the statistical analysis of microarrays.

Two methodological approaches have been followed to account for the heteroscedasticity. The first approach involves the estimation of differentially expressed genes directly from the heteroscedastic data by means of penalized \(t\)-statistics (e.g. SAM method of (Tusher et al., 2001), mixed or hierarchical Bayesian modeling (e.g. Baird et al., 2004; Hsiao et al., 2004), appropriate maximum-likelihood tests (e.g. Wang and Ethier, 2004) and, recently, gene grouping schemes (e.g. Comander et al., 2004; Delmar et al., 2005a,b). The second approach, which we follow...
in this article, involves finding appropriate transformations that stabilize the variance of the data. After variance stabilization the data can be analyzed by standard, simple and universally accepted tools, such as ANOVA models.

Section 2 outlines some existing variance stabilizing transforms that have been applied to microarray data. Section 3 proposes a new method called the Data-Driven Haar–Fisz transform for microarrays (DDHFm) and compares its performance with existing methods by means of simulated and real cDNA data in Section 4. We show that DDHFm is superior to existing methods in terms of variance stabilization and Gaussianization of the transformed intensities.

2 ESTABLISHED VARIANCE STABILIZATION METHODS

For brevity we discuss and compare the performance of different variance stabilization techniques without, at this stage, worrying about differential expression. For this reason we consider data obtained from one-color microarrays. Generalization to two-color experiments will be considered in future work.

2.1 Log-based transformations

Smyth et al. (2003) suggest using the log transform for microarray intensities. By assuming that the ‘lognormal distribution is an extremely good approximation to the bulk of the data’ (Hoyle et al., 2002) as in model (2), the log transform \( \log(Y_i) \) should stabilize the variance of the gene intensities and bring their distribution closer to the Gaussian. An extension of this approach then considers background corrected intensities, \( \hat{Z}_i = Y_i - \bar{a} \), which may be negative and cannot be handled by the simple log function. Based on this notion, several authors have studied alternative logarithmic-based transformations for microarray data.

Tukey (1977) defines the ‘Started Log’ transformation as \( \text{sLog}(Z) = \log(Z + k) \), where \( k \) is a positive constant estimated via \( k = \hat{\sigma}_Z^2/(\hat{\sigma}^2 + \hat{\sigma}_n^2) \), so that it minimizes the deviation from variance constancy. Alternatively, Holder et al. (2001) developed the log-linear hybrid transformation as Hyb1(Z) = \( Z/k + \log(k) - 1 \), for \( Z \leq k \) and Hyb2(Z) = \( \log(Z) \), for \( Z > k \). This transformation has also been called Linlog by Cui et al. (2003). As with sLog, the optimal \( k \) is estimated by \( k = \sqrt{2\hat{\sigma}_Z/\hat{\sigma}_n} \).

2.2 The generalized logarithm transformation (glog)

Munson (2001), Durbin et al. (2002) and Huber et al. (2002) independently developed the generalized logarithm transformation (referred to as glog in Rocke and Durbin, 2003). For data that come from model (1) with the mean–variance dependence (4), glog is assumed to produce symmetric transformed gene intensities with stabilized variance. The glog formula is:

\[
Z = \log\left\{ (Y - \hat{\mu}) + \sqrt{(Y - \hat{\mu})^2 + \hat{c}} \right\},
\]

where \( c \) is estimated by \( \hat{c} = \hat{\sigma}_Z^2/\hat{\sigma}_n^2 \). Rocke and Durbin (2001) described algorithms to estimate \( \alpha \) and \( c \) from one-color cDNA data. Although estimation of \( \alpha \) can be conducted without replicated genes, estimation of \( c \) involves estimation of \( \hat{\sigma}_n^2 \), which requires replication. Maximum-likelihood methods for \( c \) estimation only, based on Box and Cox (1964), were also developed by Durbin and Rocke (2003) for the case of two-colors microarrays and thus they are not relevant to the present work.

2.3 Spread-versus-level plot transformation (SVL)

Archer et al. (2004) describes a different variance stabilization approach based on plotting the log-median of the replicated intensities on the x-axis (level) against the log of their fourth-spread (a variant of the interquantile range) on the y-axis (spread). Then the estimated slope of the subsequent linear regression model fit indicates the appropriate Box–Cox power transformation.

3 DATA-DRIVEN HAAR–FISZ TRANSFORMATION FOR MICROARRAYS

This section describes how the recent DDHF transform can be adapted for use with microarray data. Our adaption requires a subtle organization of microarray intensities into a form acceptable for the DDHF transform. We call our adaption the DDHF transform for microarray data, or DDHFm.

Recently, a new class of variance stabilization transforms, generically known as Haar–Fisz (HF) transforms, were introduced by Fryzlewicz and Nason (2004). In that work the HF transform used a multiscale technique to take sequences of Poisson random variables with unknown intensities into a sequence of random variables with near constant variance and a distribution closer to normality. Later Fryzlewicz et al. (2006) introduced the DDHF transform which used a similar multiscale transform but additionally estimated the mean–variance relation as part of the process of stabilization and bringing the distribution closer to normality. See also Fryzlewicz and Delouille (2005). Hence, the DDHF transform can be used where there is a monotone mean–variance relationship but the precise form of the relationship is not known. In other words, DDHFm is ‘distribution-free’ in that the precise data distribution, such as model (1), need not be known or specified. See the Appendix section for further details on the HF and DDHF transforms.

Both the HF and DDHF transforms rely on an input sequence of positive random variables \( X \), with mean \( \mu_i \) and variance \( \sigma^2_i \) with some monotone (non-decreasing) relation between the mean and variance \( \sigma^2 = h(\mu) \). Both HF and DDHF transforms work best when the underlying \( \mu_i \) form a piecewise constant sequence. In other words, when consecutive \( \mu_i \) are often very close or actually identical in value but large jumps in value are also permitted. However, microarray data are usually not organized in this sequential form. Microarray intensities \( Y_i \) usually come in replicated blocks: i.e. \( Y_{i,r} \) is the \( r \)-th replicate for the \( i \)-th gene.

For the \( i \)-th gene what we do know is that the underlying intensity \( \mu_{i,r} \), for \( Y_{i,r} \), is identical for each replicate \( r \) (this is the reason for replication). So, if the intensities for all replicates for a given gene \( i \) were laid out into a consecutive sequence we would know that their underlying \( \mu_i \) sequence was constant.

To be able to make efficient use of the DDHF transform we would need to sort our intensities in order of increasing \( \mu_{i,r} \) so that the sequence would be as near piecewise constant as possible. In actuality as we do not know the \( \mu_i \) (since that is what we are trying to estimate) we cannot sort the sequence into increasing \( \mu \) order. So, we do the next best thing in that we order the replicate sets according to their increasing mean observed value where the mean is taken across replicates. The idea is that the observed mean estimates \( \mu_{i,r} \) and the observed mean ordering estimates the ‘correct’ true mean ordering. For example, suppose there were four replicates and four genes with observed (raw) intensities.
Variance stabilization using a multiscale approach

<table>
<thead>
<tr>
<th>Rep 1</th>
<th>Rep 2</th>
<th>Rep 3</th>
<th>Rep 4</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene 1</td>
<td>13</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>Gene 2</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>Gene 3</td>
<td>100</td>
<td>102</td>
<td>99</td>
<td>103</td>
</tr>
<tr>
<td>Gene 4</td>
<td>73</td>
<td>74</td>
<td>74</td>
<td>75</td>
</tr>
</tbody>
</table>

Then ordering these replicates according to the means of replicates for each gene (indicated in the last column), and concatenating gives a sequence of

10 11 12 11 13 12 13 14 73 74 74 75 100 102 99 103.

This ordered sequence of intensities within replicate blocks forms the input, denoted \( (X_i)_{i=1}^{1024} \) in the Appendix section, to the DDHF transform. After transformation any further technique that has been applied previously to variance stabilized and normalized data may be applied here.

4 RESULTS

Durbin et al. (2002) and Rocke and Durbin (2003) compared the performance of glog with the background-uncorrected log (Log) and the background-corrected log (bcLog) transforms. By considering 18 deterministic \( \mu \) values, each corresponding to a gene, they simulated \( Y_{r,i} \) with \( r = 1, \ldots, 1000 \) and \( i = 1, \ldots, 18 \) intensities from the two-component model (1) with parameters \( (\alpha, \sigma_q, \sigma_s) = (24800, 0.227, 4800) \) and assessed the performance of the methods in terms of the resulting transformed gene intensity variances and skewness coefficients. The two major results of Durbin et al. (2002) state that glog ‘stabilizes the asymptotic variance of micro-array data across the full range of the data, as well as making the data more symmetric’ than the other methods under comparison.

In Durbin et al. (2002) though, after simulating the intensities with the parameters mentioned above, the data were subsequently transformed using (5), with the known model parameters \( (\alpha, \sigma_q, \sigma_s) \). This procedure is biased. In practice, the true parameters are not known and have to be estimated, which results in inferior overall variance stabilization performance. Below, we demonstrate this by simulating data from the two-component model and estimating the parameters.

Additionally, in our simulations described next, we also transform our data with the background uncorrected log (Log) method, the log-linear hybrid transform, the spread-versus-level transform and our new DDHFm method. We do not use background corrected log and the started log, because both of them produce negative background corrected intensities, especially for small \( \mu \)s, and we have observed that they result in highly asymmetric data.

4.1 One-color cDNA data acquisition

We simulate from the two-component model (1) with parameters estimated from real cDNA data, obtained from the Stanford Microarray Database (http://smd.stanford.edu/). Two sets of data are considered. The first one comes from McCaffrey et al. (2004) study on mouse cDNA microarrays to investigate gene expression triggered by infection of bone marrow-derived macrophages with cytosol- and vacuole-localized Listeria monocytogenes (Lm). Each gene was replicated four times. The dataset numbers were 40 430, 40 571, 34 905 and 34 912.

The second set comes from Pauli et al. (2006) work to identify genes expressed in the intestine of Caenorhabditis elegans using cDNA microarrays. Student’s \( t \)-tests for differential expression were conducted with eight replicates for each gene. The dataset numbers were 36 590, 38 262, 38 265, 39 215, 40 157, 41 833, 41 834 and 41 886.

4.2 Simulations based on McCaffrey et al. (2004) data

We wish to simulate a likely \( \mu \) signal using our real cDNA data. As in the example of Section 3, we estimate the mean of replicates for each gene from our two datasets. These means are ordered and concatenated in a single vector from which we sample 1024 equispaced values. This sequence of sample means, shown in Figure 1, forms our simulated \( \mu \) signal (‘the truth’). This procedure is repeated for both real datasets.

From each of the 1024 \( \mu \) levels we simulate \( p = 4 \) replicated raw intensities \( Y_{r,i} \), where \( r = 1, \ldots, 4 \) and \( i = 1, \ldots, 1024 \), using the simdurbin2() function from the DDHFm package which simulates from model (1). To obtain \( Y_{r,i} \), model (1) was considered with parameters \( \alpha = 340, \sigma_q = 0.9 \) and \( \sigma_s = 95 \) as estimated (and rounded) from the McCaffrey et al. (2004) dataset. These parameters are re-estimated as in Rocke and Durbin (2001), then applied to the transformation methods that require their estimation (glog and Hyb) and the data are subsequently transformed. We iterate the above procedure \( k = 1000 \) times, and produce \( Y_{r,i} \) raw intensities, where \( r_i \) denotes the \( r \)-th replicate of the \( k \)-th iterated sequence. Finally, we concatenate the transformed \( Y_{r,i} \) into a single ‘output’ vector for each \( i \), from which we will derive our results. In other words, our output consists of 1024 output vectors \( Y_i \) of length \( p \times k = 4000 \) transformed observations.

The effectiveness of the methods is assessed in terms of adjusted SDs (\( \bar{\sigma} \)) of the replicated transformed intensities of each \( \mu \). Each \( \bar{\sigma} \) is computed as follows. The SD, \( \sigma \), of the stabilized sample of 4000 values is computed for each \( \mu \). We noted that each method stabilizes the variance to a different value. So, for each method we compute the mean of \( \sigma \)s over the whole \( \mu \) set, denoted as \( \bar{\sigma} \), and adjust each \( \sigma \) by computing \( \bar{\sigma}_r = \sigma / \bar{\sigma} \). In this way the different stabilization methods can be compared directly.

Additionally, we evaluate the Gaussianization properties of each transform by means of D’Agostino–Pearson \( K^2 \) test for normality (D’Agostino, 1974): the test is appropriate for detecting deviations...
The more concentrated the $\sigma_s$ are $\sim 1$ (the straight line in the figures), the better the stabilization has been performed. Figure 2 evidently shows the superiority of $glog^q$ over $glog^e$ for both $\sigma_q$ values, indicating the direct effect on variance stabilization when the glog parameters are being estimated. The means of the estimated parameters over the $k = 1000$ sequences were estimated as $\bar{\alpha} = 430.22$, $\bar{\sigma}_q = 0.85$ and $\bar{\sigma}_e = 104.5$. Further analysis has showed that the large difference of the estimate $\bar{\alpha}$ from $\alpha$, frequently observed over the $k$ iterations, is the main cause of the degradation in $glog^e$ performance.

Figure 3 shows Hyb and log variance stabilization results. Notice that both methods fail to stabilize the adjusted SDs of the transformed intensities and, similarly to $glog^e$, their performance depends on the $\sigma_q$ value: the smaller the $\sigma_q$ gets, the better variance stabilization is achieved. For small $\sigma_q$ though, Log seems to work better than the other two methods.

In Figure 4 we note that SVL seems to perform well, especially for small $\sigma_q$, but its performance is still inferior to DDHFm. DDHFm clearly outperforms every other method and its variance stabilization results are very similar with those of $glog^q$ (but, of course, $glog^q$ uses known parameters and cannot be used in practice).

Figures 5 and 6 show the Gaussianization results of SVL and DDHFm, which had the best variance stabilization performances. To produce the respective dotplots, we have estimated the D’Agostino-Pearson $K^2$ $p$-value for each set of transformed intensities. In the figures we present these 1024 $p$-values (dots) over the 1024 ‘mean-sorted’ genes. We interpret $p$-values >0.05 to indicate good Gaussianization and have plotted a horizontal line in the plots to aid interpretation.

We notice that SVL fails to normalize most of the transformed intensities for any $\sigma_q$. At $\sigma_q = 0.9$, DDHFm normalizes 55% of the transformed intensities but a slight downward trend is apparent, indicating that DDHFm normalization performance degrades as $\mu$ gets larger. For $\sigma_q = 0.3$, though, DDHFm normalizes the 91% of the transformed data with inexistence of a particular trend. DDHF normalizes better than SVL and outperforms the other transforms, owing to its superior variance stabilization properties.
4.3 Simulations based on Pauli et al. (2006) data

We simulate, as before, $k = 1000$ sequences from $n = 1024$ genes. Here we replicate each gene $p = 8$ times in order to show the performance of selected methods when more replicates are available. We generate the $m$ signal and then simulate raw intensities from the two component model with parameters $a = 900$, $s_e = 196$ and $s_h = 0$ derived from Pauli et al. (2006) cDNA data analysis. We compare gloge, Log, SVL and DDHFm transforms, which for small $s_h$ produced the best results in the previous section.

The top section of Table 1 shows the summary statistics of the adjusted SDs $\tilde{\sigma}$ of the transformed data for each method. Better concentration of the $\tilde{\sigma}$ around 1 suggests better variance stabilization. We observe that the best performance is achieved by DDHFm with $/C24$ 3.5 times lower range and four times lower SD from the best competitor (Log transform).

The bottom section of Table 1 shows the $K^2$ $p$-value summary statistics. Again, DDHFm performs better than any other method.

DDHFm also has the first Quantile (Q1) of its $p$-values distribution $>0.05$.

4.4 Application to real cDNA data

In this section, we transform the McCaffrey et al. (2004) real cDNA data. The need for data transformation is suggested by a preliminary analysis which indicates that the replicate SD increases with the replicate mean.

We apply DDHFm, Log, SVL and glog transforms to the dataset and compute the adjusted replicate SDs. Ideally, the five sequences of $\tilde{\sigma}$ should be as closely concentrated around 1 as possible. Figure 7 shows the variance stabilization results of the methods. Notice that DDHFm $\tilde{\sigma}$s range approximately from 0 to 3.5 (the dotted lines in the lower panel) with estimated SD of $\tilde{\sigma}$, $\tilde{\sigma} \sim 0.35$, while the best competitor glog produces $\tilde{\sigma}$s that range from 0 to 3.95 with $\tilde{\sigma}$s range from 0 to 5.8 with $\tilde{\sigma}$, $\tilde{\sigma} \sim 0.51$. Log and SVL perform worse than glog (their $\tilde{\sigma}$, $\tilde{\sigma} \sim 0.46$). Since DDHFm produces $\tilde{\sigma}$s that are more closely concentrated around one than of any of the competitors, we conclude that this is the best transformation for our dataset.

Table 1. Summary statistics of the adjusted SDs ($\tilde{\sigma}$) and $K^2$ $p$-value ($K^2$) for the various transforms

<table>
<thead>
<tr>
<th>Method</th>
<th>Min</th>
<th>Q1</th>
<th>Med</th>
<th>Q3</th>
<th>Max</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tilde{\sigma}$</td>
<td>gloge</td>
<td>0.248</td>
<td>0.666</td>
<td>1.120</td>
<td>1.335</td>
<td>1.473</td>
</tr>
<tr>
<td>Log</td>
<td>0.770</td>
<td>0.961</td>
<td>0.992</td>
<td>1.020</td>
<td>1.475</td>
<td>0.120</td>
</tr>
<tr>
<td>SVL</td>
<td>0.830</td>
<td>0.915</td>
<td>0.967</td>
<td>1.040</td>
<td>1.512</td>
<td>0.121</td>
</tr>
<tr>
<td>DDHFm</td>
<td>0.907</td>
<td>0.979</td>
<td>1.000</td>
<td>1.020</td>
<td>1.090</td>
<td>0.030</td>
</tr>
<tr>
<td>$K^2$</td>
<td>gloge</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.981</td>
</tr>
<tr>
<td>Log</td>
<td>0.000</td>
<td>0.007</td>
<td>0.165</td>
<td>0.503</td>
<td>0.998</td>
<td>—</td>
</tr>
<tr>
<td>SVL</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.150</td>
<td>0.996</td>
<td>—</td>
</tr>
<tr>
<td>DDHFm</td>
<td>0.000</td>
<td>0.085</td>
<td>0.291</td>
<td>0.597</td>
<td>0.997</td>
<td>0.995</td>
</tr>
</tbody>
</table>
5 CONCLUSIONS AND FURTHER RESEARCH

This article has introduced DDHFm, a new method for variance stabilization for replicated intensities that follow a non-decreasing mean–variance relationship. The DDHFm is self-contained and does not require any separate parameter estimation. The DDHFm is also ‘distribution-free’ in the sense that a parametric model for intensities does not need to be pre-specified. Hence, it can be used in situations where there is uncertainty about the precise underlying intensity distribution.

Simulations have shown that DDHFm not only performs very good variance stabilization but also it produces intensities that have distribution much closer to the Gaussian when compared to other established methods.

The superior performance of DDHFm combined with its ability to adapt to a wide range of distributions with non-decreasing mean–variance relationship make it an ideal tool for variance stabilization for microarray data.

This paper has not addressed the separate, but related, issue of calibration (i.e. adapting to the over location and scale of separate slides). This is an issue for DDHFm but to judge from the results on stabilization is not a significant issue. However, it would be possible to use DDHFm in conjunction with a calibration technique in a similar way to the combination of calibration and stabilization available in the vsn package described in Huber et al. (2003). We conjecture that stabilization would be again superior for DDHFm since the use of DDHFm requires somewhat more computational effort than glog type methods. Our future aim is to investigate this more challenging problem as well as develop direct Haar–Fisz methods for calibration.

ACKNOWLEDGEMENTS

E.S.M. is the grateful recipient of a Wellcome Prize Studentship awarded to G.A.R. and G.P.N. G.P.N. was partially supported by an EPSRC Advanced Research Fellowship.

Conflict of Interest: none declared.

REFERENCES


APPENDIX

THE DATA-DRIVEN HAAR–FISZ TRANSFORM

Let X = {X_i}_{i=1}^n denote an input vector to the DDHTF. The following list specifies the generic distributional properties of X.

(1) The length n of X must be a power of two. We denote J = log_2(n). In practice, if our data is not of length 2^J, then we reflect the end of our dataset in a mirror-like fashion so that the ‘padded’ sequence has a length which is a power of two.

(2) {X_i}_{i=1}^n must be a sequence of independent, non-negative random variables with finite positive means \mu_i = E(X_i) > 0 and finite positive variances \sigma_i^2 = Var(X_i) > 0.
The variance $\sigma^2_i$ must be a non-decreasing function of the mean $\rho_i$; we must have $\sigma^2_i = h(\rho_i)$, where the function $h$ is independent of $i$.

For example, let $X_i \sim \text{Pois}(\lambda_i)$. In this case, $\rho_i = \lambda_i$ and $\sigma^2_i = \lambda_i$, which yields $h(x) = x$. Naturally, in many practical situations the exact form of $h$ is unknown and needs to be estimated. Below, we describe the Haar–Fisz Transform (HFT) in the cases where $h$ is known and unknown, respectively. (For microarrays the DDHF

describe the Haar–Fisz Transform (HFT) in the cases where

input vector

'Gaussianization' of

then apply the inverse HT to obtain a vector which is closer

approximately) stabilizing the variance of

closer to normality.

We now introduce the HFT: a multiscale algorithm for (approx-

The relation $Y = F_hX$ defines a nonlinear, invertible operator $F_h$ which we call the HFT (of $X$) with link function $h$.

In practice $h$ is often unknown and needs to be estimated. Since $\sigma^2_i = h(\rho_i)$, ideally we would wish to estimate $h$ by computing the empirical variances of $X_1, X_2, \ldots$ at points $\rho_1, \rho_2, \ldots$, respectively, and then smoothing the observations to obtain an estimate of $h$. Suppose for the time being that the $\rho_i$s are known and, as an illustrative example, consider $\rho_i = \rho_{i-1}$. The empirical variance of $X_i$ can be pre-estimated, for example, as $\hat{\sigma}_i^2 = (X_i - \bar{X}_{i+1})^2/2$. Note that on any piecewise constant stretch, our pre-estimate is exactly unbiased.

The above discussion motivates the following regression setup:

$\hat{\sigma}_i^2 = h(\rho_i) + e_i,$

where $e_i = \hat{\sigma}_i^2 - \sigma_i^2 = (X_i - \bar{X}_{i+1})^2/2 - \sigma_i^2$ and 'in most cases' $\mathbb{E}(e_i) = 0$. Of course, in practice, the $\rho_i$s are not known and, since we pre-estimate the variance of $X_i$ using $X_i$ and $X_{i+1}$, it also makes sense to pre-estimate $\rho_i$ by $\tilde{\rho}_i = (X_i + X_{i+1})/2$. Note that for each $k = 1, \ldots, 2^j-1$, we have $\rho_{2k-1} = \psi_i^1$ and $\hat{\sigma}_{2k-1}^2 = 2(d_{i-1}^2)^2$, which leads to our final regression setup:

$2(d_{i-1}^2)^2 = h(\psi_i^1) + e_k. \quad (6)$

In other words, we estimate $h$ from the finest-scale Haar smooth and detail coefficients of $(X_i)_{i=1}^n$, where the smooth coefficients serve as pre-estimates of $\rho_i$ and the squared detail coefficients serve as pre-estimates of $\hat{\sigma}_i^2$.

As we restrict $h$ to be a non-decreasing function of $\rho_i$, we choose to estimate it from the regression problem Equation (6) via least-squares isotope regression, using the 'pool-adjacent-violators' algorithm described in detail in Johnstone and Silverman (2005, Section 6.3). The resulting estimate, denoted here by $\hat{h}$, is a non-decreasing, piecewise constant function of $\rho$.

The DDHFT is performed as above except that $\hat{h}$ replaces $h$. 

2553